Contenido

EL MODEL RELACIONAL

El modelo relacional es un modelo de datos basado en dos disciplinas matemáticas: la lógica de predicados y la teoría de conjuntos.

Quizás debido a este sólido fundamento teórico, que proporciona a este modelo una robustez excepcional, los SGBD relacionales (o SGBDR) son actualmente los que tienen una mayor implantación en el mercado.

El modelo relacional fue propuesto originalmente por Edgar Frank Codd en su trabajo A
1970, aunque no se implementó comercialmente Relational Model of Data for Large Shared Data Banks

hasta finales de la década.

Bacalao EF

Codd trabajaba para IBM, pero no fue esta multinacional la que creyó antes en las posibilidades del modelo relacional, sino más bien la competencia, y muy especialmente Oracle, empresa que nació justamente con el nombre de Relational Software.

1. Estructuración de los datos

El modelo relacional permite construir estructuras de datos para representar las distintas informaciones del mundo real que tengan algún interés.

Las estructuras de datos construidas siguiendo el modelo relacional están formadas por conjuntos de relaciones.

Las relaciones pueden concebirse como representaciones tabulares de los datos.

A las relaciones también se le llama informalmente tablas.

Tupla

En el ámbito de las BD, podemos definir tuple como una secuencia finita de objetos que comprende las diferentes asociaciones entre cada atributo de la relación y un valor concreto, admisible dentro del dominio respectivo.

Fila es un sinónimo tuple más informal.

Hay que precisar los siguientes extremos:

• Toda relación debe tener un nombre que la identifique unívocamente dentro de la base de datos.

• Cada fila está constituida por un tuple de datos relacionados entre sí, llamado también registro, que guarda los datos que nos interesa reflejar de un objeto concreto del mundo real.

• En cambio, cada columna contiene, en cada celda, datos de un mismo tipo, y se la puede llamar atributo o acampar.

• Cada celda, o intersección entre fila y columna, puede almacenar un único valor.

Ejemplo de relación

La tabla 1.1 refleja la estructuración tabular de la relación ALUMNO, que contiene sus datos personales correspondientes a los individuos matriculados en un centro docente.

Cada fila contiene los datos pertenecientes a un mismo alumno. La relación tiene un nombre (ALUMNO), como cada una de las columnas (DNI, Nombre, Apellidos y Teléfono). Si estos nombres son bastante significativos, permiten captar enseguida el sentido que tienen los valores de los datos almacenados en la relación.

Tabla: 1.1. Ejemplo de relación ALUMNO

DÍAS	Nom Cognoms		Teléfono
47126654F	José	Bel Rovira	453641282
51354897S	Anna	Pacheco Cuscó	723352151
56354981L	Xavier	Rius Montalvo	726922235

Toda BD relacional está formada por un conjunto de relaciones o tablas generalmente interrelacionadas entre sí que forman una especie de red.

Esta sencilla forma de visualizar la estructura de las bases de datos relacionales resulta muy comprensible para la mayoría de usuarios. Pero es necesario profundizar en algunas características adicionales de las relaciones, a fin de poder distinguir claramente los ficheros tradicionales.

1.1. Caballero

En cuanto al modelo relacional, un dominio consiste en un conjunto finito de valores indivisibles.

El dominio de un atributo o campo está formado por el conjunto de valores que el campo puede tomar.

Los atributos sólo pueden tomar los valores que estén incluidos dentro del respectivo dominio. En caso contrario no son valores válidos, y un SGBD relacional no puede permitirse su almacenamiento.

Ejemplo de dominio:

Examinamos el atributo Teléfono de la relación ALUMNO. Si lo definimos de tal modo que sólo pueda almacenar nueve caracteres (Porque los teléfonos siempre constan de nueve dígitos) de tipo numérico (ya que las letras no pueden formar parte de un número de teléfono), el dominio de este atributo incluirá todas las combinaciones posibles (109 en concreto, que es una gran magnitud, pero finita).

Otra cosa es que muchos de estos valores nunca se podrán corresponder con valores existentes en el mundo real (por ejemplo, difícilmente un operador asignará a uno de sus abonados una cadena de nueve ceros como identificador telefónico). Para conseguirlo, habría que restringir mucho más el dominio de atributo al definirlo.

Centrémonos ahora en el atributo Apellidos. Contendrá los valores de los dos apellidos de los alumnos que les tengan, separados por un espacio en blanco. Por tanto, este campo está definido para que pueda almacenar dos objetos del mundo real: primer apellido y segundo apellido.

Conceptualmente, los usuarios podrán distinguir entre los dos objetos representados y los programadores de aplicaciones podrán truncar, en caso necesario, el resultado obtenido al realizar una consulta del campo Apellidos. Pero cualquier DBMS relacional considerará el valor contenido en el atributo Apellidos para que atómica, sin ninguna estructuración interna.

Nota: No sería conveniente almacenar en un único campo los dos apellidos si queremos acceder a ellos individualmente. Si éste es el caso, lo apropiado sería tener dos campos: apellido1 y apellido2.

Debemos considerar dos tipologías de dominios:

- Dominios predefinidos. Son los tipos de datos que admita cada SGBD, como, por ejemplo (mencionados de forma genérica, ya que hay muchas especificidades en función de los diferentes sistemas gestores), las cadenas de caracteres, los números enteros, los números decimales, las fechas, etc.
- Dominios definidos por los diseñadores. Consisten en restricciones adicionales aplicadas sobre el dominio predefinido de algunos atributos, establecidas por los diseñadores y administradores de bases de datos.

Ejemplo de dominio definido por el diseñador

En una relación para almacenar los datos de los aspirantes a bombero, podría establecerse el campo IMC, para registrar los respectivos índices de masa corporal.

Pues bien, se podría restringir el dominio de este campo de tal forma que no admitiera aspirantes con valores inferiores a diecinueve ni superiores a treinta, ya que la normativa no lo permite.

1.2. Esquema y Estado (o extensión)

Toda relación consta de un esquema y su estado (o extensión).

El esquema de una relación consiste en un nombre que la identifica unívocamente dentro de la base de datos, y en el conjunto de atributos que aquélla contiene.

Es muy recomendable, para evitar confusiones en la implementación ulterior, seguir uniformemente una notación concreta a la hora de expresar los esquemas de las relaciones que forman una misma base de datos.

A continuación, se detallan las características de uno de los sistemas de notación más frecuentes:

- Es necesario escribir el nombre de las relaciones con mayúsculas y preferiblemente en singular.
- Se debe escribir el nombre de los atributos empezando con mayúscula y continuando con minúsculas, siempre que no se trate de siglas, ya que entonces es más conveniente dejar todas las letras en mayúsculas (como DNI). Con el fin de hacer los nombres compuestos más leídores, se puede encabezar cada palabra de las que forman el nombre del campo con una letra mayúscula (por ejemplo: DataNaixement, TelefonParticular, etc.).

Ejemplo de esquema de una relación

ALUMNE (DNI, Nom,Cognom1, Cognom2, Teléfono)

CP:{DNI}

VNN: {Apellido1, Apellido2}

Cabe precisar que el orden en el que nos muestran los atributos es indiferente, por definición del modelo relacional. En este esquema faltaría por añadir las restricciones de relación (se verán más adelante)

Los atributos de una relación son únicos dentro de ésta. Su nombre no puede estar repetido dentro de una misma relación. Ahora bien, distintas relaciones sí pueden contener atributos con el mismo nombre.

El esquema de una Base de datos consiste en los esquemas de todas las relaciones y restricciones de integridad definidas.

Por otra parte:

El estado de una relación consiste en los valores de los datos almacenados en todos los tuplas que ésta contiene.

Ejemplo de estado

Si tomamos como base, una vez más, la relación con esquema ALUMNO (DNI, Nombre, Apellidos, Teléfono) de la tabla 1.1, su estado sería una lista en la que figurarían todos los alumnos de la base de datos:

Alumno 1: 47126654F, José, Bel Rovira, 453.641.282

Alumno 2: 51354897S, Anna, Pacheco Cuscó, 723352151

Alumno 3:56354981L, Xavier, Ríos Montalvo, 726922235

Por tanto:

El estado de una Base de datos es el estado de todas las relaciones que la conforman.

Cualquier pequeño cambio en el estado de una de las relaciones cambia el estado de la Base de datos.

1.3. Grado y Cardinalidad de una relación

El grado de relación depende del número de atributos que incluye su esquema.

Ejemplo de grado de una relación

La relación con esquema ALUMNO (DNI, Nombre, Apellidos, Teléfono) es de grado 4, porque tiene cuatro atributos.

La cardinalidad de una relación viene dada por su número de tuplas.

Es decir, depende del estado en cada momento.

Ejemplo de cardinalidad

Si nos fijamos en la tabla 1.1, la cardinalidad de la relación ALUMNO es 3, porque su estado contiene tres tuplas correspondientes a los tres alumnos que, de momento, están matriculados.

2. Restricciones de Integridad del Modelo

2.1 Restricción de Valor No Nul

La definición de una restricción de valor no nulo sobre un conjunto K de atributos de una relación R expresa la siguiente propiedad: “no debe existir en R una tupla que tenga el valor nulo en algún atributo de K”.

Dicho de otra forma, los atributos incluidos en el conjunto K son obligatorios.

Ejemplo sobre el esquema ALUMNO:

VNN: {Nombre, Apellido}

"No debe haber en ALUMNO ningún tuple que tenga el valor nulo ni en Nombre ni en Apellidos".

2.2. Restricción de UNICIDAD

Con el fin de resultar útil, el almacenamiento de la información debe permitir la identificación de los datos. En el ámbito de las bases de datos relacionales, los tuplas de las relaciones se identifican mediante las llamadas claves.

Una clave es un subconjunto de los atributos que forman el esquema de una relación cuyos valores no se pueden repetir en tuplas diferentes, en caso de que no sean valores NULES todos ellos.

Pero una clave no debe contener atributos innecesarios que no contribuyan a la identificación inequívoca de los distintos tuplas. Las claves deben ser mínimas, tales que ningún subconjunto propio sea capaz por sí solo de identificar las tuplas de la relación.

Exemple de clau

En la tabla ALUMNO con los atributos DNI y Nombre obviamente identificaríamos a los tuplas (MAL: PERMITIRÍA QUE DOS ALUMNOS TENGAN EL MISMO DNI). Pero ésta no sería una clave porque un subconjunto de ésta, DNI, serviría para identificarlas sin necesidad de Nombre: la clave sería solo DÍAS.

Todas las claves de una relación tienen la restricción de UNICIDAD a menos que se designen como CLAVE PRIMARIA. Una Clave primaria no hace falta poner que tiene restricción de Unicidad ni de Valor No Nul porque lo lleva implícito en el concepto

Una restricción de unicidad no implica que todos los atributos que la conforman tengan Valor No Nul (VNN). Si alguno de los atributos tuviera VNN, no serían comparables.

Ejemplo de Unicidad

Sea el siguiente esquema: PERSONA(Nombre, Apellido1, Apellido2, Dirección)

UNI:{ Nombre, Apellido1, Apellido2}

VNN:{ Nombre, Apellido1}

Podría darse el siguiente estado:

Nombre	Apellido1	Cognom2	Dirección
María	Pérez	NULO	C/ Las Rosas 13
María	Pérez	NULO	Por Tarongers 2
María	Pérez	García	Av Blasco Ibáñez 100
María	Pérez	Ibáñez	Plaza Lope de Vega 30

Las dos primeras filas, al ser Apellido2 NULL, no son comparables. Pero las que tienen valor en todos los atributos que forman la unicidad no pueden tener valores iguales.

2.3. Restricción de CLAVE PRIMARIA

SÓLO HAY UNA CLAVE PRIMARIA EN CADA RELACIÓN.

La Clave primaria debe ser una Unicidad cuyos atributos sean todos No Nuls

Elección de la clave primaria:

- De entre las posibles unicidades que cumplen que además son no nulos sus atributos, es necesario elegir la CLAVE PRIMARIA. Éstas también se llaman CLAVES CANDIDATAS.
- El administrador de la BD es quien elige la clave primaria de la relación, de entre las claves candidatas disponibles. SE DEBE ELEGIR LA QUE MENOS ATRIBUTOS TENGA
- Si esta posibilidad no existe (no hay claves candidatas), es necesario añadir a la relación un atributo adicional que haga de identificador.

Por definición, el modelo relacional no admite tuplas repetidas, es decir, no permite la existencia de tuplas en una misma relación que tengan los mismos valores en cada uno de sus atributos.

Cuando hablamos de clave primaria nos referimos a la clave que, finalmente, el diseñador lógico de la base de datos elige para distinguir unívocamente a cada tuple de una relación del resto.

Todas las relaciones tendrán al menos restricción de Clau Primària.

2.4. CLAVE FORANA

También se puede llamar clave EXTERNA O AJENA.

El uso de claves ajenas es el mecanismo que proporciona el modelo relacional para expresar asociaciones entre los objetos representados en el esquema de la base de datos. Este mecanismo se define para que estas asociaciones, si se realizan, se hagan siempre adecuadamente.

Con este objetivo, se añade al esquema de una relación, R, un conjunto de atributos que hagan referencia a un conjunto de atributos de una relación S. A ese conjunto de atributos se les denomina clave ajena de la relación R que hace referencia a la relación S.

Si la tabla S (la referenciada) tiene una clave primaria formada por 3 atributos, al crear en otra tabla una clave ajena dirigida a la tabla S, la clave ajena deberá tener 3 atributos con dominios iguales o compatibles

Una clave ajena sólo puede tomar valores que sean previamente valores de clave primaria de alguna tupla de la taula a la qual va dirigida.

A esta propiedad se le denomina INTEGRIDAD REFERENCIAL.

Ejemplo Clave Forana:

S(s0, s1, s2, s3)

CP:{s0,s1,s2}

R(r0,r1,r2,r3,r4)

CP:{r0}

CAj:{r1,r2,r3}->S

EL MODEL RELACIONAL

Ejemplo relación LIBRO y AUTOR

id_lib	título	tipo	autor_id
LIB-000016	Crónica de una muerte anunciada	Novela	GAGA
LIB-000017	Siempre NO	Teatro	GAGA
LIB-000008	Doce cuentos peregrinos	Cuento	GAGA
LIB-000001	El club de los suicidas	Novela	ROST
LIB-000004	Poemas	Poesía	BERU

autor_id	nombre
GAGA	Gálamo Gante
ROST	Robert Steinball
BERU	Bertrand Rusbelt

El atributo autor_id en la tabla (o relación) LIBRO, en el actual estado, sólo puede tomar los valores: GAGA, ROST, BERU o NULL

El esquema sería el siguiente:

AUTOR(autor_id, nombre)

CP:{(autor_id}

LIBRO (id_libro, título, tipo, id_autor)

CP:{ (id_libro}

CAj:{ (autor_id}->AUTOR

Ejemplo de cómic:

PROVEEDOR (vcod: d_vcod, nombre: d_nom1, ciudad: d_ciu)

CP:{vcod}

PIEZA(zcod: d_zcod, nombre: d_nom2, color: d_color, peso: d_peso, ciudad: d_ciu)

CP:{zcod}

PROYECTO(ycod: d_ycod, nombre: d_nom3, ciudad: d_ciu)

CP: {ycod}

EL MODEL RELACIONAL

PEDIDO (vcod: d_vcod, zcod: d_zcod, ycod: d_ycod, fecha: d_fecha, canto: d_cant)

CP:{vcod, zcod, ycod, fecha}

CAj:{vcod} -> PROVEEDOR

CAj:{zcod} -> PIEZA

CAj:{ycod} -> PROYECTO

RECLAMACIÓN (cod_recl: d_rcod, vcod: d_vcod, zcod: d_zcod, ycod: d_ycod, fecha_ped: d_fecha, motivo: d_mot, fecha_recl: d_fecha)

Código de barras: {cod_recl}

CAj:{vcod, zcod, ycod, cierre_ped} -> SOLICITUD

2.4.1 Restauración de la Integridad Referencial

Ante una operación de actualización (modificación o borrado) de la base de datos que viole la integridad referencial el SGBD puede:

- Rechazar la operación
- Aceptar esa operación realizando además alguna acción compensatoria para que se cumpla la integridad referencial.
  - o poniendo valores nulos
  - o propagando la acción en cascada

Esta característica no la contemplaremos para describir el esquema. Lo veremos cuando hacemos el diseño físico (implementación en un SGBD) en la unidad de Lenguaje de Definición de Datos.

3. Restricciones de Integridad del Usuario

Restricciones de integridad del usuario son aquellas que no pueden expresarse con las propiedades anteriores (restricciones del modelo).

Hay ocasiones en las que con las restricciones del modelo no nos es posible expresar todas las reglas que deben cumplir los datos que almacenamos y es necesario introducir restricciones de usuario (diseñador). En la medida de lo posible, es necesario intentar evitarlas y expresar el máximo con las restricciones del modelo.

Éstas pueden ser:

- Restricciones de integridad estática (CREAR ASERCIÓN...).
- Restricciones de integridad de transición (Disparadoras).

Para que una base de datos sea válida, deben cumplirse todas las restricciones de integridad que ésta tenga definidas.

La comprobación de las restricciones de usuario y todas las demás restricciones del modelo (valor no nulo, unicidad, restricción de dominio, ...) es competencia del SGBD que debe asegurarse de que toda actualización de la base de datos genera un nuevo estado que satisface todas las restricciones.

Las restricciones del usuario se pueden expresar en un lenguaje formal (cálculo relacional de tuplas) pero éste no es objeto del módulo por lo que en los ejercicios se expresaron en lenguaje natural.

Ejemplo de restricción de integridad del usuario

Al dar de alta a un nuevo alumno de la relación ALUMNO, podríamos exigir al sistema que lo validara, mediante el algoritmo correspondiente, si la letra introducida del NIF se corresponde con las cifras introducidas previamente, y que denegara la inserción de lo contrario, a fin de no almacenar una situación en principio no admisible en el mundo real.
